Maratona de Programação
Semana de Informática 2025

André G. Santos Salles V. G. Magalhães
Dener V. Ribeiro Henrique C. Padula Samuel R. L. Pinto

Departamento de Informática
Universidade Federal de Viçosa (UFV), Brasil

Semana de Informática, 2025

A. 30ª Edição da SI (25/29)[1]

Resumo

- Agora em 2025 ocorre a 30ª Edição da SI
- Em 2024 ocorreu a 29ª Edição da SI
- ...
- Dado uma edição, informar o ano (e vice-versa)

Solução

- A diferença entre o ano e a edição atual é de $2025 - 30 = 1995$
- Esta diferença se mantém ao longo das edições
- Se o valor da entrada for entre 1 e 30, somar 1995, senão subtrair
  - Se o resultado for negativo ou zero, escrever 0 (não houve SI no ano)

[1] quantos times dos 25 participantes passaram a questão e número de submissões

B. Busca na Planilha (25/26)

Resumo

- Planilha para auxiliar Staff entregar balão, com colunas
  - 1 e 2: Nome e computador do time
  - 3 e 4: Marcam que o time acertou e recebeu balão da questão 1
  - 5 e 6: Marcam que o time acertou e recebeu balão da questão 2
  - 7 e 8: Marcam que o time acertou e recebeu balão da questão 3
  - ...
- Dado N, dê a coluna que marca que um time acertou a questão N

Solução

- Escrever $2N + 1$

C. Capivaristo e seu tatatatatatatatatatatatataravô (25/42)

Resumo

- Informar o ancestral de Capivaristo N níveis acima

Solução

- $1 \Rightarrow$ `pai`
- $2 \Rightarrow$ `avo`
- $3 \Rightarrow$ `bisavo`
- $4 \Rightarrow$ `trisavo`
- > 4 escrever $N - 5$ "`ta`" e completar com `tataravo`

D. Doce, doce, parece que tem mel (0/5)

Resumo

- Entrada: grafo direcionado G com peso nas arestas
- Encontrar o menor caminho $(1, V)$ em G com número par de arestas.

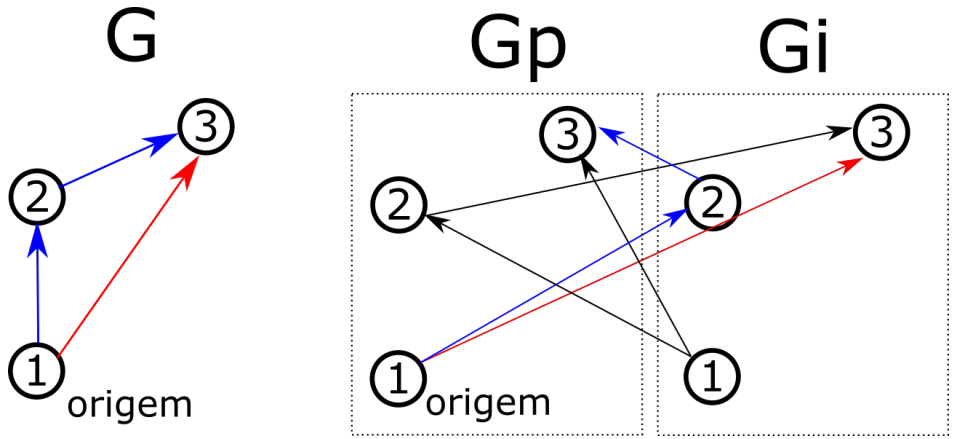
Solução 1

- Adaptar o algoritmo de Dijkstra
- Cada vértice terá dois valores "dist":
  menor caminho par até ele e menor ímpar.
- Os valores deverão ser fechados separadamente pelo algoritmo
- Ao fechar o valor par de um vértice v, atualizar o valor ímpar dos vizinhos de v (de forma análoga ao fechar valor ímpar)

D. Doce, doce, parece que tem mel (0/5)

Solução 2: *Layering*

- Criar grafo com duas cópias G_p e G_i de G (apenas os vértices).
- Para cada aresta (u, v) em G, criar aresta ligando u de G_p em v de G_i. Também criar ligando u de G_i em v de G_p.
- Os caminhos (u, v) em G_p são justamente os caminhos usando número par de arestas.
- Usar Dijkstra para encontrar caminho de 1 até V (ambos em G_p)
- Ex.: todo caminho 1 3 em G_p terá tamanho par. Toda vez que o caminho sai de G_p por uma aresta, ele voltará a G_p por mais uma.



E. Equilibrando balões (4/19)

Resumo

- Entrada: há até 15 balões com massa M_i e volume V_i
- Há um objeto com massa O_m e volume O_v
- Determinar subconjunto de balões que, juntamente com o objeto, possuem densidade igual à do ar (valor K da entrada)

Solução 1 $\mathcal{O}(2^N)$

- Testar todos subconjuntos com backtracking
- Solução ok, pois N é pequeno, $N \leq 15$ balões

Solução 2 $\mathcal{O}(N \cdot 2^N)$

- Testar todos subconjuntos com bitmask
  - os bits dos inteiros 0 a $2^N - 1$ representam todos subconjuntos possíveis
    Ex: $0000 = \emptyset$, $0001 = \{1\}$, ... $1110 = \{2, 3, 4\}$, $1111 = \{1, 2, 3, 4\}$.
- Loop de 0 a $2^N - 1$ ($\mathcal{O}(2^N)$), verificar quais bits são 1 ($\mathcal{O}(N)$)
- Solução ok para 15 balões

F. Falando Capivarês (0/3)

Resumo

- Palavras em capivarês usam apenas letras minúsculas e seguem N regras, cada uma impondo quais letras podem aparecer após outra
- Ex.: `a: a b` indica que após um 'a' podem vir apenas 'a' ou 'b'
- Dada uma palavra S, qual o mínimo de alterações de letras para ela ser válida em capivarês?

Solução – Programação Dinâmica com $|S| \times 26$ estados

- $D[i][j]$: mínimo de alterações até a i-ésima letra para ela ser letra j

$$D[i][j] = (S[i] \neq j?) + \begin{cases} \min\{D[i-1][k],\ \text{se há regra } k\text{: } j\} & \text{se } i > 0. \\ 0, & \text{se } i = 0 \end{cases}$$

  - $S[i] \neq j?$ conta 1 alteração na i-ésima letra (exceto a própria, conta 0)
  - Na 1ª letra ($i = 0$) conta 0 ou 1, dependendo se $S[0]$ for $=$ ou $\neq j$
  - Nas demais, o mínimo de alterações até alguma letra que pode vir antes
  - Retornar min $D[|S|\text{-}1][j]$, j='a'..'z' (min alterações até fim da palavra)

F. Falando Capivarês (0/3)

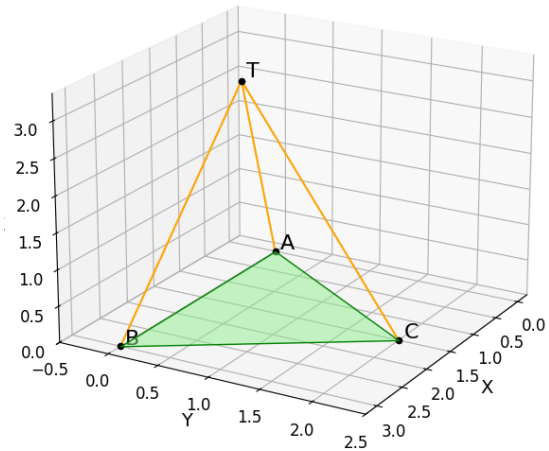
Exemplo de implementação bottom-up $\mathcal{O}(N|S|)$

- $D[i][j]$: mínimo de alterações até a i-ésima letra para ela ser j
  (inicialmente todos com valor alto, pelo menos $|S|+1$)
- $D[0][j] = 1$ para todo $j =$ 'a'...'z', exceto $D[0][S[0]] = 0$
  (para a 1ª letra, precisa 1 alteração; exceto a própria, que não precisa)
- Para cada posição i, da primeira à penúltima
  - Para cada letra j de 'a' a 'z'
    - Para cada letra k que pode aparecer depois de j
      $D[i+1][k] = \min\{D[i+1][k], D[i][j] + (S[i] == k ? 0 : 1)\}$
      ($D[i+1][k]$ usará o menor $D[i][j]$ tal que exista regra $j: k$)
- Retornar menor valor da última posição $(\min D[|S|-1][j], j =$ 'a'...'z')

G. Guardião da lagoa (0/1)

Resumo

- Tenda montada com 3 hastes de tamanho L apoiadas nos pontos A, B, C do plano
- Determinar a altura h da tenda (z de T)



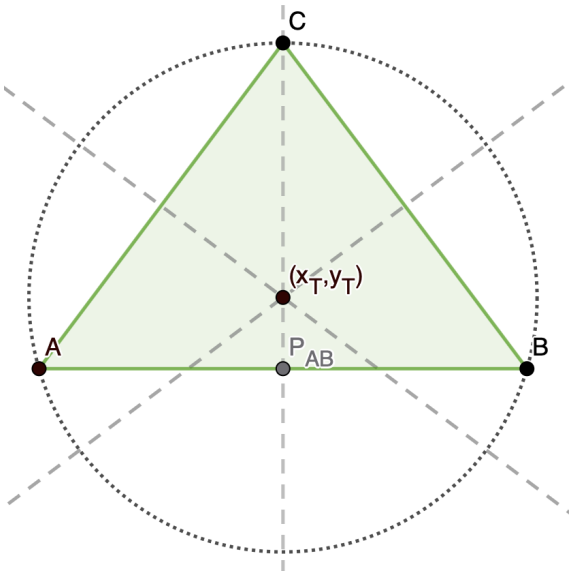
Solução 1, por geometria

- A projeção do topo T no plano ABC é o circuncentro do Δ ABC
- O circuncentro de um Δ é o ponto de encontro de suas mediatrizes
- Mediatriz é a reta perpendicular que passa pelo pt. médio de um lado *(cont...)*

G. Guardião da lagoa (0/1)

Solução 1 *(cont...)*

- Ponto médio do lado AB:
  $P_{AB} = ((x_A + x_B)/2, (y_A + y_B)/2)$
- Reta que passa por AB tem coeficiente
  $m = (y_A - y_B)/(x_A - x_B)$
- Mediatriz é sua perpendicular,
  coeficiente $-1/m$, passa pelo centro:
  $(y_{P_{AB}} - y_T) = -\frac{1}{m}(x_{P_{AB}} - x_T)$



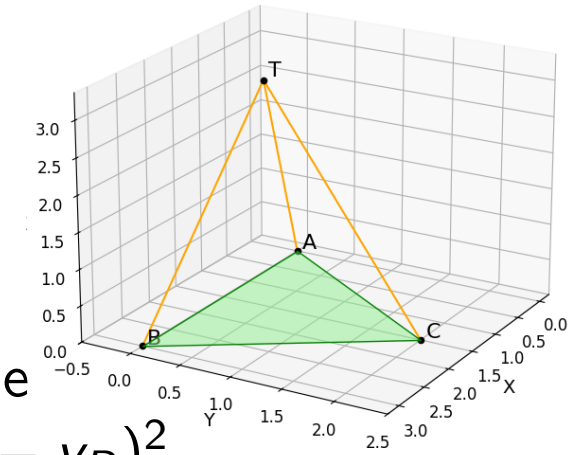
(cuidado! evite escolher dois pontos que causam divisão por 0)

- O mesmo vale para lados AC e BC e, com duas dessas equações, resolve-se um sistema linear para as incógnitas x_T e y_T
- Por fim, h pode ser encontrado pela distância de uma das bases a T:
  $L^2 = (x_T - x_A)^2 + (y_T - y_A)^2 + h^2$

G. Guardião da lagoa (0/1)

Solução 2, mais simples, por distância euclidiana

- Sabe-se que a distância de A a T é L:
  $L^2 = (x_T - x_A)^2 + (y_T - y_A)^2 + h^2$
- Equações semelhantes
  para distância de B a T e de C a T



- Das equações de A a T e de B a T, temos que
  $(x_T - x_A)^2 + (y_T - y_A)^2 = (x_T - x_B)^2 + (y_T - y_B)^2$
- Equações semelhantes são obtidas para A e C e para B a C
- Sistema linear com 2 dessas equações e 2 incógnitas x_T, y_T
  obs: a eq. acima não parece linear, mas termos quadráticos são cancelados
- Encontrados x_T e y_T, obtém-se h pela equação da distância

H. Hipopotomonstrosesquipedaliofobia (3/9)

Resumo

- Encontrar a maior sequência de letras numa palavra sem que nenhuma apareça mais de K vezes
- $K \leq 1000$ e palavra pode ser gigante! (10^5 letras)

Solução força bruta: $\mathcal{O}(KN)$ – TLE

- O tamanho máximo que não repete mais que K é $26K$
- Loop ninhado: contar as letras em cada grupo de $26K$ caracteres

Solução força bruta melhorada: $\mathcal{O}(KN)$ – TLE

- Ao testar cada grupo, parar assim que uma letra tiver $> K$ ocorrências

H. Hipopotomonstrosesquipedaliofobia (3/9)

Solução *two pointers*: $\mathcal{O}(N)$

- Dois índices, Esq e Dir, para início e final da sequência verificada
- Um map ou um vetor de contadores para cada letra a-z
- Inicialmente tudo 0, inclusive maxtam para o maior tamanho
- Repetir até Dir chegar ao final
  - Mover Dir para direita enquanto for válido:
    Dir++, contar a letra S[Dir], atualizar maxtam se for o caso
    - Interromper se ocorrências $> K$ (basta verificar contador de S[Dir])
  - Mover Esq para direita enquanto for inválido
    descontar a letrar S[Esq], Esq++
    - Interromper se toda ocorrência $\leq K$ (basta verificar contador de S[Dir])
- Retornar maxtam

I. Invocando Oberlan Romao (0/1)

Resumo

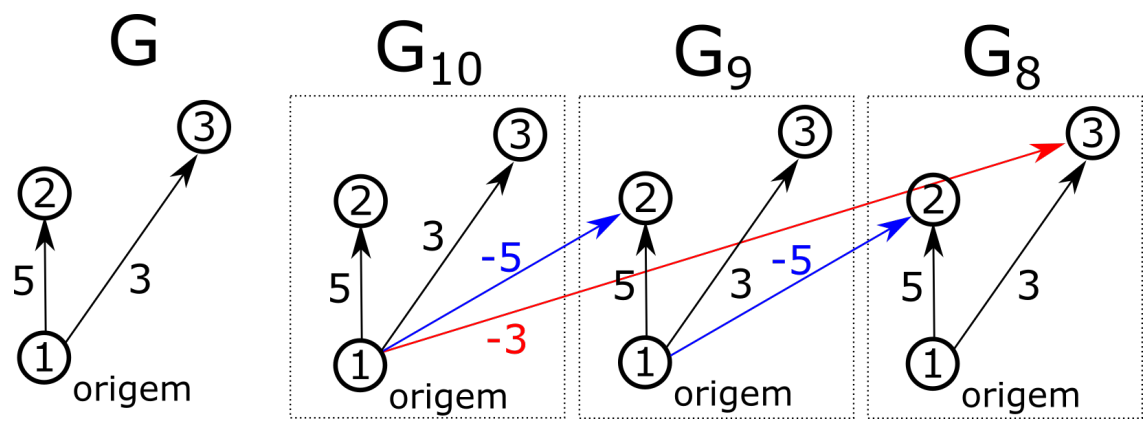
- Grafo direcionado com pesos e valor inicial de Mana (até 15)
- Cada aresta tem o consumo de combustível
- Achar caminho de menor consumo
- Capivaristo consegue negativar consumo na aresta usando uma quantidade de mana (variável por aresta)

Solução

- *Layering*, similar à apresentada para o problema D
- Criar camadas onde G_m representa o estado do grafo com m de mana sobrando
- Para aresta (u, v, C, P) (C: combustível gasto e P: mana necessária para negativar)
  - criar arestas (u, v, C) em todas camadas
  - criar resta $(u, v, -C)$ de u da camada k a v da camada $k - P$
- Usar Bellman-Ford para encontrar caminho mínimo

I. Invocando Oberlan Romao (0/1)

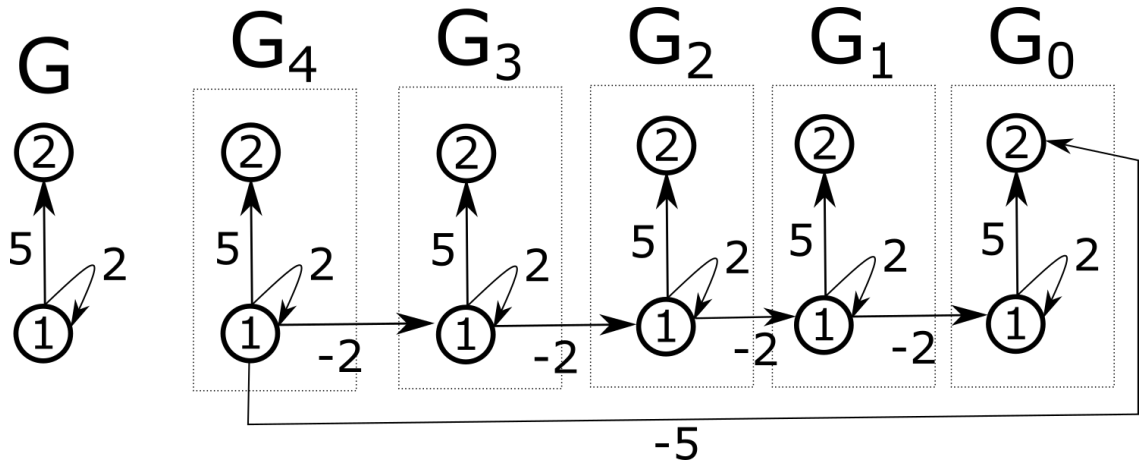
- Exemplo:
  - aresta $(1, 2)$ de consumo 5 precisa de 1 mana para ser negativada
  - aresta $(1, 3)$ de consumo 3 precisa de 2 mana para ser negativada
- Com 10 de mana posso percorrer $(1, 3)$ gastando 3 de combustível; ou -3 (nesse caso, iria para um estado com 8 de mana restantes)
- Não há arestas ligando uma camada a outra com mais manas.



(obs.: por questão de espaço, são mostradas apenas 3 camadas)

I. Invocando Oberlan Romao (0/1)

- Exemplo 2:
  - aresta $(1, 2)$ de consumo 5 precisa de 4 mana para ser negativada
  - aresta $(1, 1)$ de consumo 2 precisa de 1 mana para ser negativada
- Abaixo o grafo construído para o caso de começar com 4 de mana



J. Jogada de mestre  (2/18)

Resumo

- N questões de treino da maratona para dividir entre dois amigos
- Há questões relacionadas, de tópicos ou técnicas muito semelhantes
- Na divisão, nenhum dos amigos pode ficar com questões relacionadas
- Pode-se adicionar uma única relação para impossibilitar a divisão?

Solução

- Grafo com V sendo questões e A indicando questões relacionadas
  $\Rightarrow$ a divisão é possível se e somente se o grafo for bipartido
- Para impedir a divisão, basta ligar dois vértices da mesma partição
- Cuidado! O grafo pode ser desconexo, considerar cada componente

K. Kaprekar (23/46)

Resumo

- Constante 6174 sempre pode ser obtida a partir de números de até 4 dígitos, com pelo menos 2 únicos, da seguinte forma:
  - Ordena-se os dígitos em ordem decrescente e crescente (ex.: $4231 \rightarrow 4321$ e 1234).
  - Subtrai-se o maior número do menor (ex: $4321 - 1234 = 3087$).
  - Repete-se o processo até obter 6174
- Objetivo: calcular quantos passos são necessários para se obter 6174

Solução

- Simulação (poucas iterações)
- Cuidado que o número da entrada pode ser o próprio 6174
- Cuidado com o fato de poderem aparecer números com menos dígitos (completar com 0 durante a simulação)

L. Limpando a string (11/48)

Resumo

- Strings S e T
- Verificar se pode remover porção contígua de S para igualá-la à T

Solução gulosa $\mathcal{O}(N)$

- Percorrer S enquanto puder: `while (S[i]==T[i]) i++;`
- Nesse ponto, faltam $|T| - i$ caracteres
- Percorrer S de trás para frente para verificar se os últimos $|T| - i$ caracteres são iguais aos de T

M. Minha senha super secreta (25/30)

Resumo

- Entrada: uma palavra sem vogais repetidas
- Saída: a palavra seguida das posições das vogais, em ordem alfabética
- Ex.: `PERNAMBUCO` $\Rightarrow$ `PERNAMBUCO4197`

Solução 1

- Percorrer a palavra guardando a posição de cada vogal
- Escrever as posições na ordem

Solução 2

- Percorrer a palavra e imprimir a posição do '`A`'
- Percorrer a palavra e imprimir a posição do '`E`'
- ...
- Percorrer a palavra e imprimir a posição do '`U`'